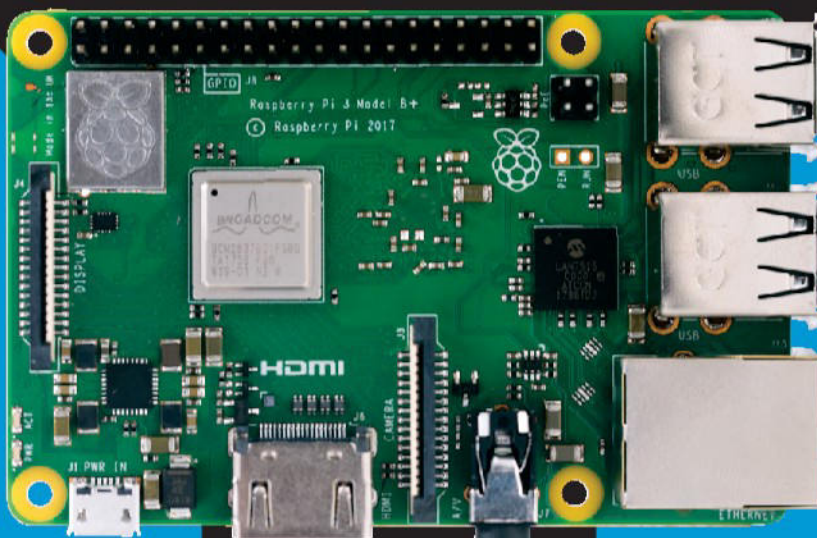
Numero 78 | Febbraio 2019 | magpi.cc raspberryitaly.com

La rivista ufficiale Raspberry Pi tradotta in italiano da RaspberryItaly

GIOCHI ARCADE

COSTRUIRE
Un flipper

PROGRAMMARE
Un gioco 3D

GIOCARE
Giochi PC

The MagPi magazine is published by Raspberry Pi (Trading) Ltd., Mount Pleasant House, Cambridge, CB3 0RN. ISSN: 2051-9982.

GIOCHI ARCADE

COSTRUIRE
Un Flipper

PROGRAMMARE
Un Gioco 3D

GIOCARE
Giochi PC

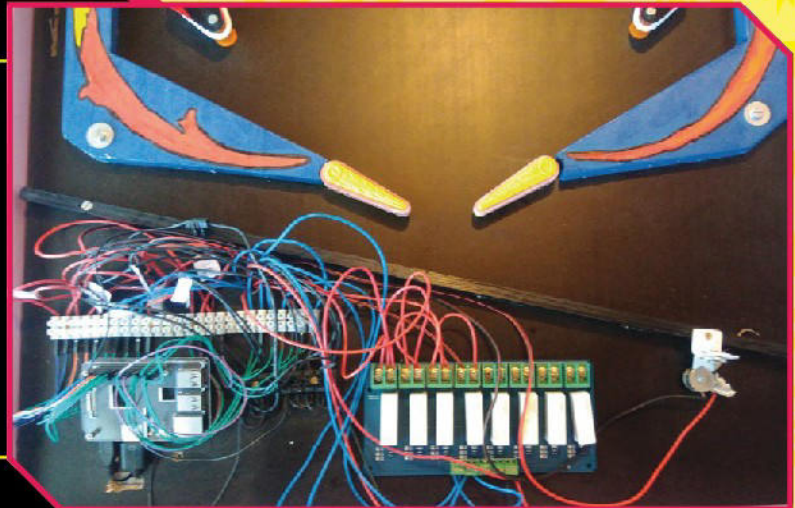
Prova questi fantastici progetti di gioco con il tuo Raspberry Pi

Ci sono molti, molti modi per utilizzare Raspberry Pi per giocare. Che tu stia costruendo un gioco fisico nel mondo reale, programmando il tuo gioco o semplicemente facendo girare dei giochi sul tuo Raspberry Pi, è uno straordinario componente hardware sia per i video games che per i giochi fisici.

Abbiamo riunito tre progetti e di ognuno ti daremo un assaggio. Per prima cosa, ti mostriamo come costruire un flipper partendo da un vecchio letto per bambini. Poi, come far girare Steam su Raspberry Pi. Infine, come programmare il tuo gioco usando Pygame Zero.

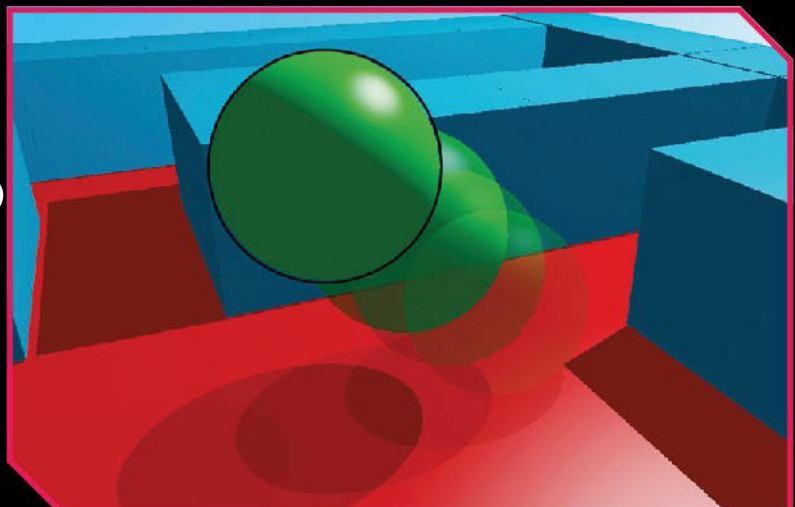
Su, andiamo a divertirci.

COSTRUIRE UN FLIPPER

PAGINA 30

LANCIARE I GIOCHI STEAM

PAGINA 38

PROGRAMMA IL TUO GIOCO PERSONALE

PAGINA 42

FLIPPER WIZARD

Quando la figlia di **Martin Kauss** è diventata troppo grande per il suo letto da principessa, lui lo ha trasformato in un flipper musicale e danzante. Ecco come puoi fare lo stesso

I bambini, man mano che crescono, lasciano dietro di loro una serie di scarti, ma ci vuole un particolare tipo di maker per vedere un vecchio lettino decorato e immaginarlo come il tavolo di un flipper.

Abbiamo parlato con Martin Kauss e gli abbiamo chiesto come abbia trasformato il letto di sua figlia in un Flipper, e abbiamo creato una guida passo-passo per aiutarti a costruirlo.

Cosa ti ha indotto a farlo?

Volevo un progetto educativo che mi insegnasse come far interagire il GPIO con il mondo reale.

Inoltre, l'obiettivo era fare qualcosa che fosse fonte di ispirazione per i miei bambini, divertente da costruire e con cui giocare.

Come sviluppatore di software con esperienza nella meccanica, volevo costruire qualcosa di fisico. Sopratutto, non avevo molta esperienza in elettronica quindi il progetto è stata una introduzione a questa materia.

K.G. Orphanides

KG. è una scrittrice, sviluppatrice e maker, sempre sbalordita dalle incredibili creazioni che escono dalla comunità Raspberry Pi.

Quanto tempo ha impiegato finora?
In totale, direi circa tre mesi. incluse preparazione, ordinazione, lettura, apprendimento, costruzione, sviluppo, test e ottimizzazione.

Quanta pianificazione avanzata hai fatto?
Non così tanta, ad essere onesti. Ho fatto piccoli passi e ho fatto molte ricerche, prototipi e test preliminari per ogni componente.

Appena ho capito il funzionamento di una parte (ad esempio il meccanismo della paletta dei flipper), ho testato le parti meccaniche fino a quando non ho ottenuto il corretto funzionamento e poi lo ho unito allo sviluppo del software, in combinazione con i montaggi e cablaggi necessari e infine il test funzionale.

Per me è stato importante concentrarmi su un componente finché lo stesso non funzionasse correttamente e solo dopo passare oltre.

Qual è stata la parte più difficile?
Non avevo esperienza con altri flipper che non fossero quelli con cui giocavo da bambino, quindi la parte più difficile è stata trovare il giusto vocabolario per poter condurre ricerche atte a comprendere il funzionamento dei singoli componenti come le palette del flipper.

La meccanica delle palette non è semplice, oltre che pericolosa, anche con un sistema (rispetto ai veri flipper) a bassa tensione come il mio - circa 36V di alimentazione. Ho bruciato una resistenza durante i lavori. E per bruciato intendo fiamme vere... anche se piccole e di breve durata.

Cosa pianificherai dopo?
Anche se il Raspberry Pi ha molti pin GPIO me ne sono rimasti pochissimi liberi. Posso solo aggiungere un altro bumper o alcuni bersagli. La decisione non è ancora stata presa. Ma il tavolo da gioco ha bisogno di altri componenti, come i respingenti.

Quantità	Componente	Prezzo unit.	Totale (Euro)
3	Palette	3 €	9 €
1	Kit anelli in gomma palette	30,73 €	30,73 €
1	Gruppo lanciatore	30,73 €	30,73 €
2	Gemma per tasto flipper	1,85 €	5,55 €
2	Tasto flipper da pannello	5,14 €	10,28 €
1	Copertura respingente (destro)	10,17 €	10,17 €
1	Copertura respingente (sinistro)	10,17 €	10,17 €
2	Pulsante flipper	14,18 €	28,37 €
3	Microswitch rollover	4,32 €	12,32 €
1	Microswitch	4,32 €	4,32 €
1	Gruppo spinner	28,77 €	28,77 €
1	Gruppo palette completo con bobina Fl-11630, finecorsa normalmente aperto (dx e sx)	50,36 €	50,36 €
15	Piolini	1,02 €	15,3 €
2	Funghetti respingenti	42,35 €	84,7 €
1	100pz guaina termorestringente	4,88 €	4,88 €
1	5m cavo multicolore 0.75 mm²	12,64 €	12,64 €
1	Alimentatore 36 V (TDK-Lamda LS150-36)	44,88 €	44,88 €
1	Cavo maschio-femmina 20cm ×40	5,4 €	5,4 €
1	2m striscia LED	9,24 €	9,24 €
1	Alimentatore 5V (TOOGOO(R) AC 110 V/220 V)	13,21 €	13,21 €
1	Modulo relè stato solido SainSmart 8 canali 5V	14,42 €	14,42 €
1	5V-220V 5A relè stato solido SainSmart 2 canali DC-DC	12,62 €	12,62 €

TOTALE: 435,29 €

Poiché questo progetto si basa sulla creazione di un flipper inedito, la tua lista dei componenti non sarà identica a quella di Martin Kauss. Tuttavia, questo elenco ti aiuterà a capire quali componenti avrai bisogno per la tua realizzazione, e quale sarà il loro costo. I prezzi (indicativi) sono riportati in euro.

I componenti specialistici del flipper arrivano a costare circa 400€ in totale, ma non sono compresi gli attrezzi, le parti per costruire il tavolo stesso e qualche componente extra per Pi e GPIO.

In questi sono inclusi un telaio - il letto da principessa della figlia di Martin, in questo caso; un pezzo di compensato per il tavolo stesso e vari pezzi di legno e alluminio per la costruzione di parti come la pista di lancio, i bersagli e le gambe del tavolo; un paio di piedini in plastica con regolazione a vite per le gambe anteriori del tavolo e un sacco di viti, dadi, bulloni e staffe di montaggio per tenere insieme il tutto.

Dal lato elettronica e componenti meccanici, Martin dice che sono da includere anche lampadine a LED, connettori (a vite / a incastro), capicorda, cavi GPIO, molle e molle di trazione (ad esempio per chiudere la pista di lancio).

Infine, avrai bisogno di una ben equipaggiata cassetta degli attrezzi, compreso un saldatore e una matassa di filo di stagno per saldatura, un voltmetro, pinza spelafili, cacciaviti, sega, un trapano e relative punte, un martello, una lima, e idealmente una terza mano con pinze a coccodrillo per tenere fermi i componenti elettronici mentre li assembli.

Rivenditori Utili

- flipperteile.de
- flipperservice.de
- pinballlife.com
- sainsmart.com
- pinball.co.uk
- pinparts.co.uk

CREA IL TUO FLIPPER PERSONALE

Questa guida passo-passo illustra le varie fasi di costruzione del flipper Principessa. Anche se il tuo flipper potrebbe essere molto diverso, i componenti chiave e le tecniche si applicano ad un'ampia gamma di costruzioni.

Ogni flipper avrà bisogno di un lanciatore, palette, bumper e bersagli. I suggerimenti – come l'utilizzo delle gambe regolabili per aiutarti ad ottenere l'angolo perfetto del tavolo – valgono per molti stili di costruzione.

Allo stesso modo, la configurazione delle connessioni GPIO di Martin Kauss e il software per gestire le luci, i sensori, i suoni e i punteggi del suo flipper sono strumenti potenti per la costruzione di qualsiasi flipper.

01 Impostare il software

Inizia con un Pi ed un'installazione pulita di Raspbian. Avrai bisogno di una connessione ad internet e ti semplificherai la vita se colleghi un monitor, che utilizzerai anche in seguito per tenere traccia del punteggio del giocatore, una tastiera e un mouse.

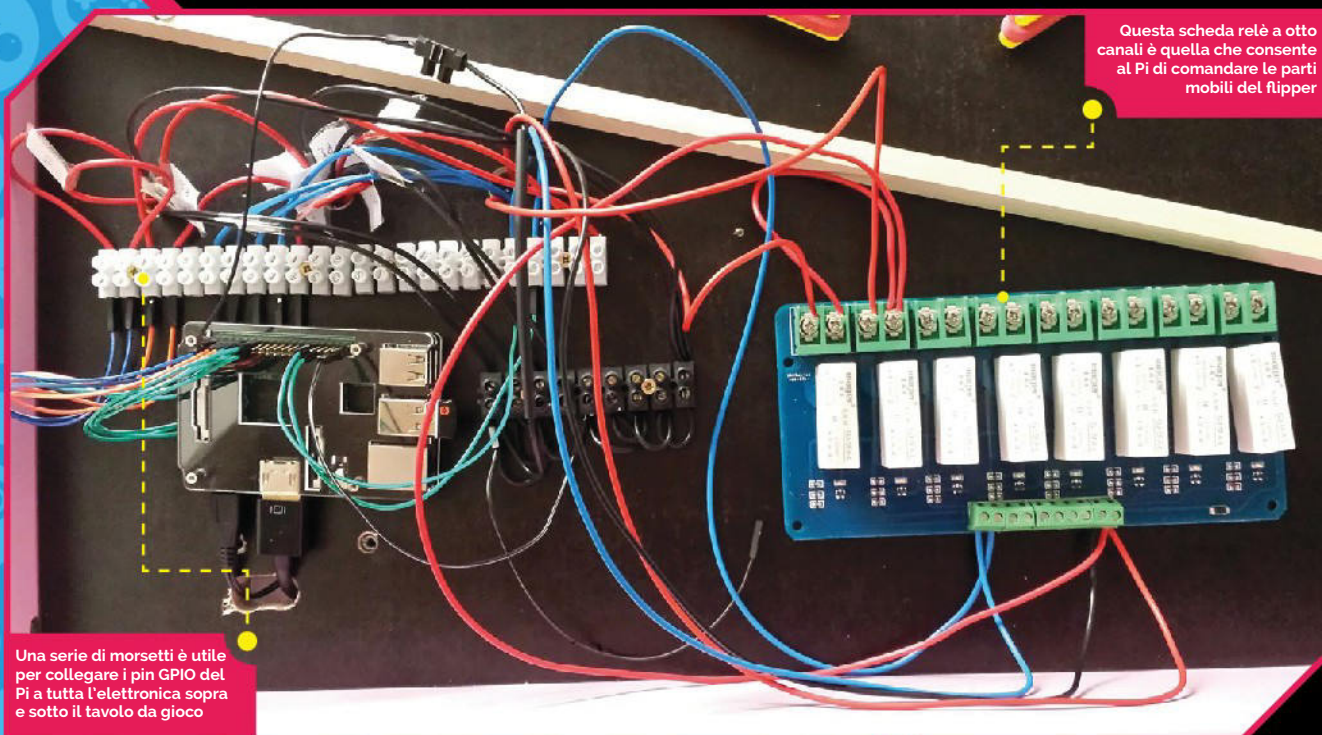
Apri una finestra del Terminale e digita:

```
sudo apt install python-pygame
git clone https://github.com/bishoph/pinball.git
cd pinball
python pinball_machine.py
```

Premendo Q si esce dal programma. I caratteri non sono inclusi, quindi per eseguire il programma devi trovare i file dei font TrueType 9σ7' &55h=+ * (86 α(+Ah=+ e copiarli in ᵃ><;ᵃ58(&5ᵃ<. & *ᵃ+87=<ᵃ – entrambi sono freewaree e disponibili online.

02 Vedere

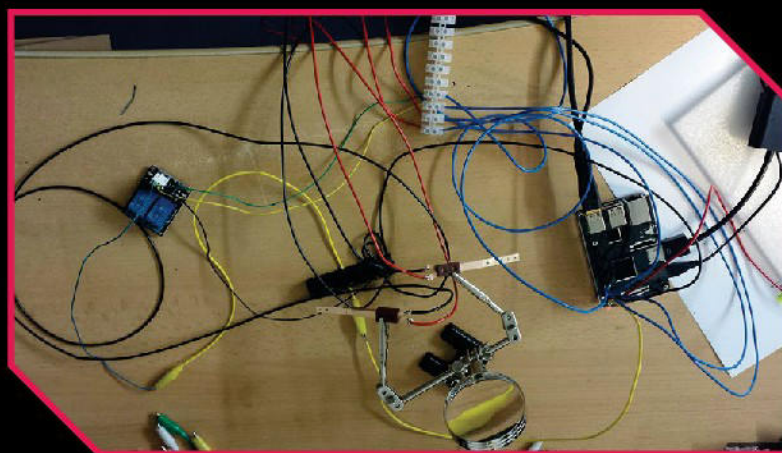
Impostare tutto è molto più facile se hai un monitor collegato al Pi, ma anche gli

Questa scheda relè a otto canali è quella che consente al Pi di comandare le parti mobili del flipper

Una serie di morsetti è utile per collegare i pin GPIO del Pi a tutta l'elettronica sopra e sotto il tavolo da gioco

Paletta sinistra: le palette sono la parte più complessa da cablare e il loro circuito include un diodo Zener per la regolazione della tensione

Come mostrato in questa anteprima di cablaggio del flipper, un relè a due canali, attivato dai pin GPIO del Pi, è dedicato a attivare le bobine del flipper

script Python del nostro flipper possono fare uso di un display esterno. Lo useremo per mostrare il punteggio attuale del giocatore, il numero di palle ancora da giocare, il punteggio più alto del flipper e qualche divertente effetto visivo.

Se ti senti in grado, puoi usare una staffa di montaggio VESA per fissare il monitor a un pannello o supporto fissati al tavolo; o, se stai lavorando con un telaio da letto come Martin, cerca lo spazio per fissarlo sulla testata del letto.

03 Affrontare la musica

È possibile riprodurre dei suoni tramite l'altoparlante integrato di un monitor o collegando degli altoparlanti all'uscita audio da 3,5 mm del Pi. I suoni sono definiti nello script **effects.py**, installato nel Passo 01. Dovrai procurarti i file audio - Martin ne ha presi alcuni da **freesfx.co.uk**.

Inseriscili nella directory **/home/pi/pinball/sounds** e modifica **effects.py** di conseguenza. Lo script attiva i suoni quando il flipper viene alimentato, quando la palla esce dalla pista di lancio, quando cade fuori dal piano di gioco, e quando innesca gli spinner o i bumper.

I campioni sonori del banco chiamato S2 vengono attivati come eventi casuali quando il flipper è in pausa.

04 Telaio e fortuna

Il flipper di Martin Kauss è nato da un letto per bambini, decorato con le immagini colorate di una principessa Disney, ma potresti anche costruirti un telaio in legno, comprare un kit base da flipper, o anche creane uno con K'nex o con il Meccano.

Il telaio misura 145 × 77 cm e per il playfield - la superficie del flipper, dove si svolge il gioco - Martin ha usato un pezzo di compensato di 230 mm con finitura nera.

Sotto il campo di gioco, ci sarà spazio per il cablaggio e gli alimentatori.

Prova prima

Assicurati che i componenti e le connessioni funzionino correttamente prima di avvitarli in modo permanente al pannello.

05 Più potenza, Igor!

Questa realizzazione richiede sia un alimentatore da 5 V, per alimentare le luci, sia un alimentatore da 36 V per le bobine utilizzate per alimentare i respingenti ed i bumper, che hanno requisiti relativamente elevati di tensione.

Ad esempio, anche se la bobina del bumper è alimentata a 36 V, il LED integrato, così come quelli sul playfield, sono alimentati dalla linea a 5 V. Per facilitare le connessioni, sul connettore GPIO del Pi è montato un connettore a saldare.

Da questo, i cavi GPIO vanno verso una serie di connettori. Questi includono sia segnali di input che i segnali di output e tutti i componenti come LED e bobine sono cablati tramite schede relè. I relè sono alimentati dal Raspberry Pi (tramite i pin 2 e 6) e l'intervento di ciascuno avviene tramite il rispettivo pin GPIO di uscita.

È utile avvitare una piccola ciabatta elettrica nella parte posteriore o inferiore del flipper, così avrai alimentazioni separate per monitor e Pi.

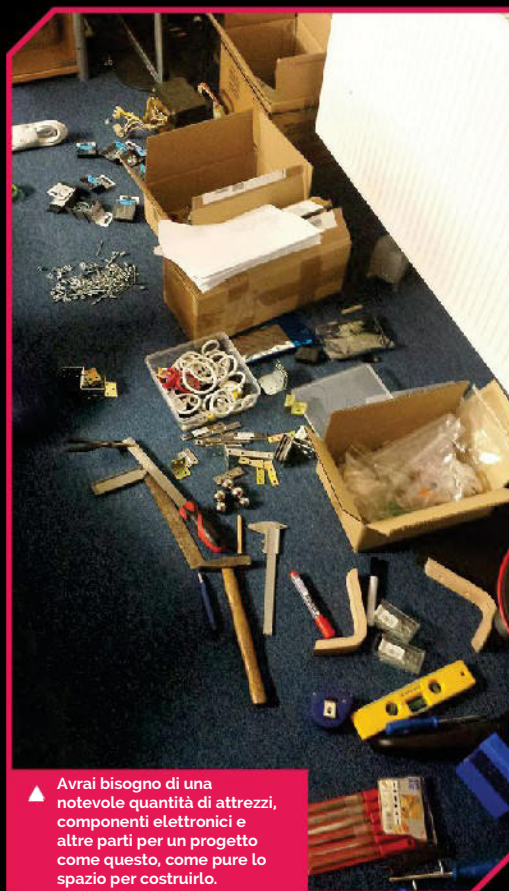
06 Tilt! Tilt! Tilt!

I tavoli dei flipper non possono essere piatti, ma ottenere l'angolo corretto per garantire che la palla rotoli verso il basso alla giusta velocità può essere difficile. Delle gambe regolabili sono molto utili per cambiare facilmente la pendenza del tavolo.

In questo caso sono state usate quattro stecche di legno quadrate, misura 4.5×4.5×100 cm per le gambe anteriori e 4.5×4.5×110 cm per le gambe posteriori. Le gambe anteriori sono state rese regolabili mediante l'inserimento di piedini di plastica che possono essere sollevati e abbassati, consentendo di effettuare alcune prove durante la costruzione.

07 Piano, disegni & fori

Una volta che hai il telaio e un playfield, è il momento di delineare le loro caratteristiche. Misura con attenzione, esegui test di posizionamento, fotografa e disegna intorno alle palette, al lanciatore, ai respingenti, e ai

Spinner: lo spinner in sé è passivo, ma sattiva un microswitch ogni volta che la palla lo capovolge, attivando il punteggio, il suono, le luci.

Avrai bisogno di una notevole quantità di attrezzi, componenti elettronici e altre parti per un progetto come questo, come pure lo spazio per costruirlo.

bumper che desideri includere. Vorrai anche piazzare i pulsanti laterali delle palette.

Oltre a questi componenti, dovrai usare lamelle di legno e di metallo per le corsie, l'area di fuga della pallina verso la corsia di lancio dopo averla persa, e la parte superiore ricurva del tavolo. Dopo aver segnato e controllato due volte la corretta posizione dei componenti, potrai effettuare i fori che ti serviranno per avvitare e cablare i tuoi componenti.

> "Ricordati di lasciare abbastanza spazio sotto al tavolo per l'elettronica"

Ricordati di lasciare abbastanza spazio sotto il tavolo per l'elettronica, incluso il Pi!

08 Assemblare lo stantuffo

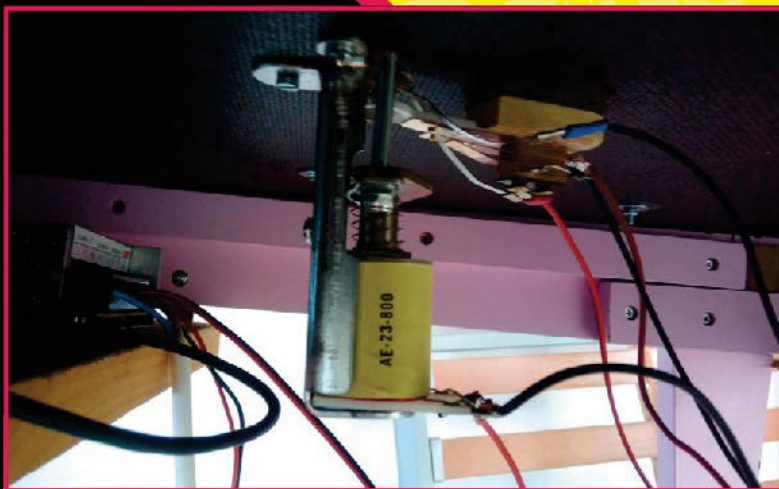
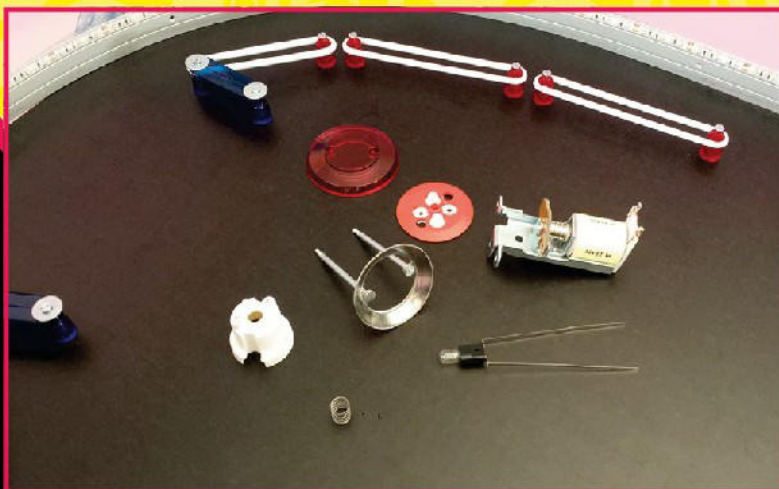
Lo stantuffo, noto anche come lanciatore a molla, è il primo componente del tuo flipper critico da assemblare. Puoi comprare un kit che include tutte le sue parti (asta, molla, alloggiamento e componenti esterni come la placca).

Verrà montato sulla destra del tavolo, con il pomello e le parti esterne sporgenti dalla parte anteriore del flipper. Una stecca di legno di lunghezza adeguata costituisce la corsia lungo la quale la palla passa dallo scarico posto sotto le palette, verso il gruppo lanciatore. Uno sportellino, tenuto chiuso da una molla di adeguata tensione, chiude la corsia di lancio finché la palla lanciata dallo stantuffo supera la velocità adatta ad aprirlo, per poi entrare nel tavolo da gioco.

09 Mi vedono rotolare

La maggior parte delle corsie e strutture di questo progetto, è costituita da lamelle di legno e di alluminio piegate e tenute in forma da blocchetti di legno avvitati al tavolo, il che rende la costruzione piuttosto facile. Ma tu vuoi usare il Pi per sapere quando la palla transita in queste corsie.

Per questo avrai bisogno di microswitch a rotolamento (rollover) che puoi trovare dai fornitori specializzati in parti di flipper, e dovrai inserirli dalla parte inferiore del tavolo. Esegui un taglio longitudinale nel playfield usando una fresa o un trapano e una lima. Monta lo switch su una tavoletta di legno, e posizionalo sotto al taglio in modo che sporga abbastanza per essere innescato dal peso della palla mentre ci transita sopra.

Questo flipper ha uno switch rollover al termine della corsia di lancio e su tutte le corsie che

▲ I funghetti respingenti sono assemblati con la loro bobina sotto al tavolo, e la "gonna" e il cappuccio, sopra

Vai online per più dettagli

Per altre informazioni e per il blog del costruttore del flipper Principessa, visita il sito web di Martin Kauss all'indirizzo **magpi.cc/iimYKq**

portano la palla fuori dal campo, più uno alla fine, giusto prima che la palla rientri nella corsia di lancio, cosicché il flipper sappia sempre quando la palla lascia il campo di gioco.

Sono tutti connessi al connettore GPIO del Pi – tramite connettori – così possono attivare gli eventi come le luci, i suoni e il punteggio.

10 Muovi le palette

Le palette, in un flipper, sono i componenti più importanti. Avrai bisogno di un alimentatore da 36 V e 5 A per far scattare le palette tramite le bobine, e di un relè a due canali per attivarle quando richiesto dal GPIO del Pi.

Poiché stiamo utilizzando il Pi per controllare il tutto, abbiamo bisogno di un kit di assemblaggio delle palette di tipo moderno, con pulsante normalmente aperto (NO) e finecorsa (EOS).

Le bobine sono montate sotto al tavolo, sotto le palette. Per ogni paletta, devi collegare il pulsante sul lato del mobile ad un pin del GPIO, il finecorsa (EOS) ad un altro, e altri due pin – tramite relè – alla bobina della paletta, che in realtà è formata da due avvolgimenti nello stesso corpo.

La prima bobina, HIGH, ha una bassa resistenza e fa muovere la paletta forte e velocemente, mentre il secondo, HOLD, ha una alta resistenza e ti permette di tenere le palette alzate quando mantieni premuti i pulsanti delle palette.

Pin	Terminale connettore	Tipo	Descrizione	Connessione Relè
3	1	IN	Pulsante paletta destra	
5	2	IN	Pulsante paletta sinistra	
8	3	IN	Finecorsa paletta destra	
10	4	IN	Finecorsa paletta sinistra	
7	5	IN	Micorswitch Spinner	
11	6	OUT	Paletta destra HIGH	Relè #1,1
12	7	OUT	Paletta destra HOLD	Relè #1,2
13	8	OUT	Paletta sinistra HIGH	Relè #2,1
15	9	OUT	Paletta sinistra HOLD	Relè #2,2
16	10	IN	Microswitch lanciatore	
18	11	IN		
19	12	IN		
21	13	IN		
22	14	IN	Switch Bumper #1	
23	15	IN	Switch Bumper #2	
2	16	IN	Switch respingenti	
26	17	OUT		
29	18	OUT		
31	19	OUT		
32	20	OUT	Luce #1, corsia di lancio	Relè #3,1
33	21	OUT	Luce #2, respingenti	Relè #3,2
35	22	OUT	Luce #3, bumper	Relè #3,3
36	23	OUT	Luce #4, bumper	Relè #3,4
37	24	IN	Microswitch pista di lancio, un segnale!	
38	25	IN	Bobina Bumper #1	Relè #4,1
40	26	IN	Bobina Bumper #2	Relè #4,2

Il flipper Principessa ha dimostrato di piacere molto ai figli di Martin e a tutti i loro amici, con la richiesta di una futura modalità multi-ball

11 Dentro i respingenti

I respingenti sono componenti a forma di cuneo appena sopra le palette, ci aiutano a far rimbalzare la palla verso di esse, durante la sua corsa sul tavolo.

Sono colorati e pieni di piolini – supporti di plastica colorata circondati da un anello di gomma sul quale la palla può rimbalzare. Un sensore a microswitch rileva quando la palla lo colpisce, innescando l'aumento dei punti, rumori e luci.

Un paio di vivaci copri-respingenti completano il tutto, e una corsia – fatta in legno – corre dietro di loro per fornire un percorso alla palla verso le palette.

12 Aggiungere un bumper o due

I bumper a forma di fungo sono tra gli elementi più iconici di un flipper. Sono disponibili in kit completi e normalmente rilevano, tramite un microswitch integrato, quando la palla colpisce la loro "gonna" di plastica. Una bobina abbassa l'asta del bumper e il gruppo dell'anello per buttare via la palla.

Puoi usare la base del bumper per segnare dove questo vada montato, con la bobina montata sotto la superficie del tavolo. Le bobine sono alimentate dai 36 V dell'alimentatore e connesse al Pi tramite la scheda relè a otto canali.

13 Aggiungere uno spinner, collegato con morsetti a vite

Uno spinner, o bersaglio rotante, è un componente classico di un flipper, con la sua corsia e un microswitch che attiva le luci e aumenta il punteggio del giocatore che lo colpisce.

Qui la corsia viene creata una rotaia metallica di alluminio. Lo spinner è montato utilizzando staffe di metallo da negozio fai-da-te e bulloni, oltre a piolini e anelli di gomma da reperire dalle ditte fornitrici di flipper.

Un microswitch con attuatore a filo è connesso con lo spinner per riconoscere il passaggio della pallina e incrementare, di conseguenza, il punteggio, azionare le luci e i suoni.

14 Luci! Azione!

Un flipper non è niente senza delle luci lampeggianti. Questo progetto usa una striscia a LED di 2 metri collegata ai 5 V dell'alimentatore per aggiungere una combinazione attraente di colori alla pista di lancio iniziale e all'"orbit"– la parte superiore ricurva della striscia di alluminio che aiuta a mantenere la palla in rotazione intorno al tavolo.

Ci sono anche luci singole nei due bumper e nei respingenti. Per un totale di quattro collegamenti al GPIO sul Pi. Lo script Python attiverà le luci quando la palla entra nel campo di gioco, colpisce i bumper, attiva lo spinner, o passa sopra gli switch di rotolamento. Le luci sono attivate dallo script anche in maniera casuale se non ci sono input o durante le pause del gioco.

Attenzione alle dita

Non toccare le bobine o le parti mobili durante i test, perché la rapida contrazione del solenoide può causare un doloroso pizzicotto

I GIOCHI PC SUL RASPBERRY PI

Trasforma il tuo Raspberry Pi in una macchina Steam Link e fai lo streaming di giochi di alta qualità da Steam su PC

Controlla, prima di giocare, di avere una buona connessione a Steam in esecuzione su un computer desktop e che il tuo controller sia ben impostato

Fai clic sull'icona Inizia a giocare per aprire Steam sul tuo PC Windows e accedere alla tua libreria di giochi

Cosa Serve

- Raspberry Pi 3B+ / 3B
- Raspbian Stretch
- Tastiera e mouse per il setup
- Un PC Windows
- Steam (e un account Steam)
- Un controller compatibile (vedi riquadro)

Steam è un negozio online zeppo di tutti gli ultimi giochi per PC, e ha recentemente annunciato il supporto per Steam Link per Raspberry Pi.

Con Steam Link, puoi eseguire lo streaming dei giochi Windows da un PC al tuo Raspberry Pi. Collega un gamepad e hai una macchina da gioco che rivaleggia con qualsiasi console.

Puoi usare praticamente qualsiasi gamepad con il Raspberry Pi (vedi 'Controller compatibili') e puoi giocare a giochi per PC di alta qualità su qualsiasi televisione o monitor in casa.

Funziona in coppia al software Steam su un PC Windows, quindi avrai bisogno di un account Steam Store e di un gioco dal negozio Steam (esistono molti giochi e demo da giocare gratuitamente).

Diamo un'occhiata a come trasformare un Raspberry Pi in una console animata da Steam Link e giocarci i giochi Windows.

Lucy Hattersley

Lucy è il direttore di The MagPi e adora gli RPG e i giochi d'avventura. Era davvero sorpresa di come Euro Truck Simulator 2 si è rilevato divertente, e guiderà un camion virtuale intorno a Cambridge di continuo.
magpi.cc

01 Impostare Steam

Inizia aprendo e accedendo a Steam sul tuo PC Windows. Se non hai Steam installato, puoi scaricarlo da **store.steampowered.com**. Fai clic su Installa Steam per scaricare il file

SteamSetup.exe nella tua cartella di download. Fai doppio clic e segui il processo di installazione di Steam. Clicca su Crea un nuovo account e segui la procedura di creazione dell'account. Infine, accedi al programma.

Avrai bisogno di un gioco, per giocare. Steam, fortunatamente, ha una grande selezione di software e demo giocabili da provare gratuitamente. Fai clic su Giochi e scegli Free to Play per vederne una selezione. Abbiamo scelto di installare la demo di Euro Truck Simulator 2.

02 Installare Steam Link

Con Steam attivo e funzionante sul PC Windows, è ora di trasformare il tuo Raspberry Pi in una Steam streaming box. Inizia con una nuova installazione di Raspbian Stretch (vedi **magpi.cc/quickstart** se hai bisogno di una rinfrescata su come installare Raspbian). Assicurati che il tuo Raspberry Pi sia collegato a Internet (idealmente con connessione Ethernet) e apri una finestra del Terminale. Digita i seguenti comandi:

```
sudo apt update
sudo apt install steamlink
```

Rispondi "y" a tutte le domande.

03 Collegare un controller

Collegheremo un gamepad PlayStation DualShock 4 al Raspberry Pi usando il Bluetooth.

Puoi usare una vasta selezione di altri controller, tra cui quelli di Xbox One, Nintendo Switch Pro e lo Steam Controller (insieme a molti altri gamepad Bluetooth). Oppure puoi collegare una tastiera e un mouse, in molti giochi. Ma un controller è il modo migliore se vuoi usare il tuo Raspberry Pi con un televisore.

Tieni la tastiera o il mouse collegati al Raspberry Pi mentre effettui le impostazioni. Clicca sull'icona Bluetooth nella barra dei menu e seleziona Attiva Bluetooth. Ora fai nuovamente clic sull'icona Bluetooth e scegli Aggiungi dispositivo.

Metti il gamepad in modalità di paring. Questo processo è diverso su ogni gamepad. Su un controller DualShock, tieni premuti i tasti PS4 e Stream sul gamepad fino a che la luce nella parte posteriore inizia a lampeggiare.

Dovresti vedere Wireless Controller nella finestra Aggiungi dispositivo sul Raspberry Pi. Selezionalo e fai clic su Accoppia. Dovresti vedere "Accoppiamento avvenuto con successo"; in caso contrario, prova a fare clic su Bluetooth> Wireless Controller> Connetti.

Ci sono voluti alcuni tentativi per avere il gamepad PS4 connesso al Raspberry Pi. Insisti.

▲ Guidando intorno alla nostra base di Cambridge in Euro Truck Simulator 2, in esecuzione su un Raspberry Pi

04 Lancia Steam

Clicca su Menu> Giochi> Steam Link per aprire l'app (ora puoi chiudere il Terminale). L'applicazione Steam Link si aprirà nella modalità a schermo intero. Dovresti vedere tre cose: il nome del tuo computer (nel nostro caso 'DESKTOPE1QNN21'), 'PS4 Controller' (o il nome del tuo controller) 'e' Connessione buona'.

Per prima cosa, testiamo la connessione. Usa il gamepad per fare clic su Impostazioni e scegliere 'Streaming e Test di rete'. La finestra Testing di rete apparirà. Si spera tu veda 'Test di rete Completo" e " Fantastico! La tua rete sembra funzionare bene con Steam Link'. Steam consiglia di utilizzare una connessione Ethernet, ma se stai tentando di utilizzare il WiFi puoi provare a spostare il tuo Pi più vicino al router. Clicca OK e usa l'icona Indietro in alto a sinistra per tornare alla finestra del menu principale.

05 Cominciare a giocare

Fai clic sull'icona Inizia a giocare in Steam Link. Dovresti vedere un codice PIN di quattro cifre in Steam Link sul tuo Raspberry Pi. Inserire il PIN nella finestra di dialogo Autorizza Dispositivo in Steam sul tuo PC Windows e fare clic su OK.

Ad un certo punto potresti vedere un messaggio di avviso "Impossibile connettersi",

Connessione Ethernet

Steam consiglia di utilizzare un cavo Ethernet per connettere il tuo Raspberry Pi al router di rete. Noi abbiamo trovato che lo streaming dei giochi va bene, ma alcuni menu erano un po' lenti. Le prestazioni miglioreranno, col tempo.

Forum Steam Link

Unisciti al forum Steam Link per ottenere risposte a qualche tua domanda. C'è un'intera sezione su Raspberry Pi e la risoluzione dei problemi.
magpi.cc/xJopzO

assieme a un messaggio 'Lo streaming richiede l'installazione di un driver aggiuntivo'.
Fare clic su Installa per scaricare e installare il driver necessario. Clicca ancora su Inizia a giocare per entrare su Steam Link.

06 Benvenuto su Steam

Ora vedrai la finestra di Steam Link 'Benvenuto su Steam'. Nella parte superiore, ci sarà una serie di icone: Web, Negozio, Libreria, Comunità e Chat. Sopra ci sono tre icone più piccole per Download, impostazioni e Power. Dovresti poter utilizzare il D-pad sul gamepad della PS4 per navigare tra le icone. Premi il tasto X per selezionare gli elementi e O per tornare indietro.

07 Impostare il controller

Il supporto di base per il gamepad funziona, ma per un supporto migliore è necessario configurare il controller. Fai clic sull'icona Impostazioni in alto a destra e scegli Impostazioni del controller. Spostati in basso fino a Configuratore PlayStation e fai nuovamente clic per inserire un segno di spunta nella casella. Apparirà la finestra 'Personalizza il tuo controller'. Lascia i valori di default, per ora. Scegli Conferma. Premi O per tornare alla finestra principale.

08 Vedi la Libreria

Seleziona Libreria nella finestra Benvenuti su Steam per visualizzare tutti i giochi della tua collezione Steam. Scegli installati e Euro Truck Simulator 2 Demo. Verrà visualizzato un avviso giallo che dice "Configurazione controller richiesta". Anche se il tuo gamepad PS4 è configurato in Steam, potrebbe richiedere una configurazione specifica per ogni singolo gioco (molti giochi Steam sono progettati con in mente una tastiera completa e mouse). Vedremo questo più tardi; per ora, fai clic su Gioca.

09 Configurare il controller

Siccome questo è il tuo primo lancio, la prima volta apparirà la selezione della configurazione. Fai clic su Sfoglia altre Config. Scegli Comunità e vedrai un bel po' di configurazioni create da altre persone. Sulla destra avranno un numero di voti. Scegli quello con il numero più alto. Noi abbiamo visto 'zaka.ahoa2' con quattro voti positivi. Premi il tasto quadrato per applicare la configurazione e avviare il gioco.

10 Scegli i tuoi controlli

Ora puoi giocare usando il trackpad PS4 per cominciare a guidare. Quando inizi a guidare, il D-pad o lo stick analogico sinistro controllerà il movimento e puoi guardarti intorno

L'interfaccia di Steam Big Picture è stata riconfigurata per funzionare sullo schermo di un televisore con un controller

Anche con il controller già collegato a Steam Link, riceverai avvisi per configurarlo (per personalizzare i controlli) per molti giochi

usando lo stick analogico destro. Porta il tuo camion a fare un giro (o gioca a qualsiasi altro gioco tu voglia). Mentre è meglio, per iniziare, utilizzare i controlli creati dalla comunità, è possibile aggiungere i tuoi controlli per ogni gioco, personalizzati secondo le tue preferenze.

11 Controlli personalizzati

Premi il pulsante PlayStation per accedere alla finestra delle impostazioni. Qui troverai la Configurazione del controller. Lì, scegli Controlli personalizzati. Seleziona il pulsante sinistro a grilletto e usa la tastiera virtuale per impostarlo su '['; quindi imposta il pulsante destro a grilletto su ']'. Ora, mentre guidi, puoi usare i pulsanti a grilletto sinistro e destro da usare per i tuoi indicatori di direzione.
Molti giochi per PC hanno controlli complessi e potresti non essere in grado di giocare a tutti i giochi usando solo il gamepad. Ma molti altri sono perfettamente giocabili con il solo controller. Divertiti a giocare ai giochi per PC sul tuo Raspberry Pi.

12 Giocare

Ora puoi andare avanti e giocare. Assicurati che i controlli funzionino correttamente e puoi spostare il tuo Raspberry Pi dall'ambiente di test, al suo posto, sotto il tuo televisione. Un modo rapido per farlo è aggiungerlo al file **.bash_aliases**. Apri una finestra del Terminale e immetti (non dimenticare il '.' prima di 'bash'):

```
sudo nano .bash_aliases
```

Dovrebbe aprire un file vuoto. Scrivi:

```
steamlink
```

Premi *GXVP>[* per salvare il file, e *GXVP>* per uscire dall'editor di testo Nano. Ora riavvia il tuo Raspberry Pi (`sudo shutdown -r now`). Quando lo riavvii, dovrebbe partire direttamente in Steam Link.

CONTROLLER COMPATIBLI

Usa questi gamepad per giocare. Per un elenco completo dei controller supportati e delle caratteristiche, vedi: **magpi.cc/BoqAxE**

PS4 DualShock 4

Il controller PS4 DualShock 4 (58 €) è una scelta popolare ed è quello che usiamo nel nostro tutorial Con un D-pad e due stick analogici, è dotato di un touchpad per replicare l'input del mouse.

Controller Xbox

Il controller Xbox (58 €) funziona in modalità wireless con la stessa impostazione del controller PS4. Benché non sia così supportato come il DualShock 4, potrai usarlo per giocare.

Controller Nintendo Switch Pro

È una scelta costosa, ma se ne hai uno a portata di mano, il controller dello Switch Nintendo (69 €) funziona bene con Steam Link. Manca però il touchpad del DualShock 4.

Controller Steam

Steam ha venduto il suo controller assieme alle Steam Machine ufficiali (che il Raspberry Pi ora sostituisce). Puoi ancora trovare un controller Steam (£ 40). Accanto a uno stick analogico ci sono due trackpad progettati per replicare gli input del mouse. magpi.cc/oBzbjt

Parte 01

PROGRAMMARE UN GIOCO ISOMETRICO: AMAZEBALLS

Pygame Zero in 3D. Certo, è possibile e ti mostreremo come in questo tutorial in tre parti per un gioco labirinto

In questa serie, fino ad ora, abbiamo imparato come usare Pygame Zero per creare dei giochi rapidamente. In questo tutorial in tre parti, useremo diverse nuove tecniche per creare un gioco di labirinti in 3D. Lo stile grafico che useremo viene chiamato 3D isometrico. Significa che il nostro display sarà fatto di cubi regolari con una leggera falsa prospettiva, perché il display è effettivamente costituito da immagini 2D. Questa tecnica è stata utilizzata in molti giochi anni ottanta come Knight Lore, Alien 8 e la mia serie, ArcVenture. In questa prima parte, costruiremo la mappa di base utilizzando liste di dati e creeremo un personaggio palla rimbalzante per far muovere il giocatore dentro al labirinto.

01 Benvenuto nella terza dimensione

Pygame Zero non è stato scritto avendo in mente i giochi 3D, ma a volte devi solo spingere i confini un po' più in là e vedere quanto lontano arriva la tua idea. Questo metodo per disegnare un'area di gioco non è in senso stretto un display 3D, ma appare come 3D e possiamo fare molto con questa tecnica. Faremo una mappa dai dati di una lista, e la costruiremo con dei cubi. Quindi inseriremo una palla rimbalzante per far muovere il giocatore nell'area di gioco tramite la tastiera. Faremo partire la palla da un lato del labirinto e quando il giocatore la farà uscire dall'altro lato, il gioco sarà completato

02 Realizzare la mappa

Come al solito in questa serie, partiamo importando il modulo `pgzrun` e non dimenticare

La palla parte da un lato del labirinto e il giocatore deve guidarla dall'altro lato

Il giocatore è rappresentato da una palla rimbalzante

I muri del labirinto sono stati creati disegnando dei cubi

Mark Vanstone

Autore di software educativo degli anni '90, autore della serie ArcVenture, scomparso nella landa desolata del software aziendale. Salvato dal Raspberry Pi!
magpi.cc/YiZnxL | @mindexplorers

di richiamare `pgzrun.go()` alla fine del codice. Le dimensioni predefinite della finestra dovrebbero essere adatte ai nostri scopi. Faremo una lista bidimensionale di numeri che rappresenteranno la nostra mappa. Ogni numero rappresenterà un quadrato sulla mappa. Realizzeremo la mappa di dodici quadrati di larghezza e dodici di altezza. Vedi **figure1.py** per come definiamo questa lista, che chiameremo `mapData`. L'altra variabile che vedi, chiamata `mapInfo`, definisce la larghezza e l'altezza della nostra mappa in una struttura chiamata dizionario.

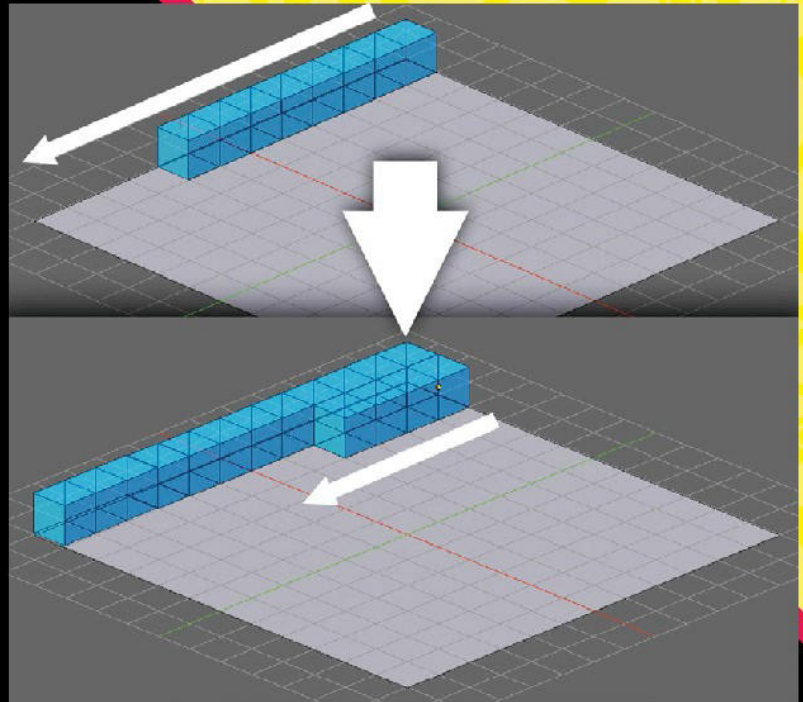
03 A come ape

Un dizionario in Python è una struttura dati molto utile. Anziché limitarci a memorizzare un elenco di valori o stringhe, possiamo dare a ciascuno di questi valori un'etichetta. Guardando il nostro dizionario `mapinfo`, possiamo vedere che ci sono due valori dichiarati; il primo ha l'etichetta 'width' e il secondo 'height'. Per estrarre i valori, dobbiamo solo scrivere `mapInfo["width"]`, che, in questo caso, ci ritornerebbe il valore 12. I dizionari sono una struttura molto utile per raccogliere assieme diverse variabili associate tra loro e possono essere identificate dalle loro etichette.

04 Mappare la mappa

Ora che abbiamo dei dati per la nostra mappa, abbiamo bisogno di definire alcune immagini che utilizzeremo per disegnarla. Possiamo fare un nuovo elenco di blocchi mappa scrivendo `mapBlocks = ["map1c","map2c"]`. Questo definirà che quando vediamo uno 0 nella lista `mapData`, significa disegna la prima immagine nella lista, che è map1c. Se vediamo un 1 nei dati, significa quindi utilizza l'immagine map2c. Per il momento, ci limiteremo a due diversi blocchi mappa. Il primo rappresenterà il pavimento; il secondo, i muri.

05 Mostrare la mappa

Il prossimo passo è tradurre i nostri dati mappa in una mappa visuale che possiamo mostrare sullo schermo. Imposteremo le basi nella nostra funzione `draw()` di Pygame Zero e quindi richiameremo una funzione `drawMap()` che scriveremo per fare effettivamente il lavoro. In **figure2.py**, vedrai che riempiamo prima lo schermo di nero, e poi disegnamo la nostra mappa. La funzione `drawMap()`, anche se breve, può sembrare un po' complicata, quindi analizziamola con calma in modo che tu capisca come funziona.

▲ La costruzione avviene da dietro a davanti, una riga alla volta, per produrre l'effetto 3D

▼ Il nostro framework di base Pygame Zero e i dati che definiscono la nostra mappa

figure1.py

SCARICA IL CODICE COMPLETO: magpi.cc/DVNCvv

› Linguaggio: **Python**

```
001. import pgzrun
002.
003. mapData = [[1,1,1,0,1,1,1,1,1,1,1,1],
004.            [1,0,0,0,0,0,0,0,0,0,0,1],
005.            [1,1,1,1,1,1,1,0,1,1,1,1],
006.            [1,0,0,0,0,0,0,0,0,0,0,1],
007.            [1,1,1,1,1,1,1,1,0,0,0,1],
008.            [1,0,0,0,0,0,0,1,0,1,1,1],
009.            [1,0,1,0,1,1,0,1,0,0,0,1],
010.            [1,0,1,0,1,0,0,1,1,1,0,1],
011.            [1,0,1,0,1,0,0,0,0,0,0,1],
012.            [1,1,1,0,1,1,1,1,1,1,1,1],
013.            [1,0,0,0,0,0,0,0,0,0,0,1],
014.            [1,1,1,1,1,1,1,1,0,1,1,1]]
015.
016. mapInfo = {"width":12, "height":12}
017.
018. pgzrun.go()
019.
```

figure2.py

```
001. OFFSETX = 368
002. OFFSETY = 200
003. mapBlocks = ["map1c","map2c"]
004. mapHeight = [0,32]
005.
006. def draw(): # Funzione draw Pygame Zero
007.     screen.fill((0, 0, 0))
008.     drawMap()
009.
010. def drawMap():
011.     for x in range(0, 12):
012.         for y in range(0, 12):
013.             screen.blit(mapBlocks[mapData[x][y]], ((x*32)-
(y*32)+OFFSETX,
014.                             (y*16)+(x*16)+OFFSETY -
mapHeight[mapData[x][y]]))
015.
```

06 Realizzare i blocchi

Utilizzeremo un ciclo annidato per scorrere i nostri dati e visualizzare la mappa. Il primo ciclo è per la direzione x e dentro di esso abbiamo un altro ciclo per la direzione y. X e y, in questo caso, non si riferiscono a coordinate in pixel dello schermo ma ai blocchi di dati. Ora immagina che stiamo trasformando i nostri dati a 45 gradi in modo da disegnare i nostri blocchi in diagonale sullo schermo. Lo facciamo con un po' di matematica per tradurre le posizioni x e y nei nostri dati in posizioni sullo schermo.

07 Una fantastica vista

Per disegnare ogni blocco, usiamo la funzione Pygame Zero `screen.blit()`, che carica una immagine e la disegna sullo schermo alle coordinate specificate. Il primo parametro di questa funzione è

▲ I rudimenti della funzione di disegno della mappa

▼ Questa tecnica di mappa era usata con diversi giochi 8 bit ed è stata ancora utilizzata per le prime versioni di Sim City e Age of Empires

il nome dell'immagine. lo otteniamo prendendo il valore di `mapData[x][y]` e usando `mapBlocks` per dare un nome all'immagine. Ora calcoliamo le coordinate dello schermo. La larghezza di ogni blocco è di 64 pixel, ma siccome stiamo disegnando in diagonale, vogliamo solo spostare di 32 pixel lateralmente ogni blocco così che si sovrappongano l'un l'altro. Verso il basso, spostiamo ogni blocco di 16 pixel.

08 Un diamante è per sempre

Per trovare la coordinata x dello schermo, moltiplichiamo il valore x dei dati per 32 e lo sottraiamo dal valore del dato y 32 volte. Questo ci dà coordinate x dello schermo che partono da 0. Poi aggiungiamo un offset predefinito per spostare il blocco iniziale al centro dello schermo. Facciamo lo stesso per ottenere la coordinata y dello schermo, ma usando un moltiplicatore di 16 per i dati x e y, aggiungono un offset per spostare il blocco iniziale un poco più in basso sullo schermo, tenendo conto che alcuni blocchi sono più alti di altri, utilizzando `mapHeight`. Quando viene lanciato, vedremo dodici righe e dodici colonne disegnate a forma di diamante.

> “Creeremo un dizionario per contenere tutti i dati che ci serve sapere del giocatore”

09 Palle

Ora è il momento di realizzare il nostro personaggio, che in questo caso sarà rappresentato da un palla rimbalzante. Inizieremo posizionando la palla e spostando la mappa. Creeremo un dizionario per contenere tutti i dati che ci serve conoscere del giocatore. Vedi **figure3.py** per osservare come definiamo i dati del giocatore. I valori `x` e `y` sono la posizione del giocatore nella mappa dati del blocco. Faremo partire il giocatore nella colonna(`x`) 0 e riga(`y`). Il valore di `frame` ci dirà quale frame della animazione mostrare. I valori `sx` e `sy` sono l coordinate effettive dello schermo dove verrà disegnata la palla.

10 Usare il blitter sul giocatore

Lasciando da parte per il momento gli altri valori del nostro dizionario giocatore, l'altra parte di codice in **figure3.py** entra nei nostri cicli `drawMap()` dopo aver blitterato i blocchi. Il codice dice "se il blocco che stiamo attualmente occupando è il blocco su cui c'è il giocatore, calcola le coordinate dello schermo (usando lo stesso calcolo dei blocchi) e disegna la palla allo schermo". Lavoriamo sulle coordinate su schermo della palla solo una volta, visto che ora faremo una funzione `doMove()` che gestirà le coordinate su schermo del giocatore da qui.

11 In movimento

La nostra funzione `doMove()` ci introdurrà a un paio di nuove tecniche Python e a altri dati ancora del dizionario giocatore. Passiamo a `doMove()` tre parametri. Il primo è la struttura del dizionario giocatore (in modo che possiamo leggere e scrivere i valori del giocatore), e poi i cambiamenti di `x` e `y` che vogliamo fare nelle unità blocchi. X e y saranno 0, 1 o -1 e rappresentano un movimento nei nostri blocchi di dati della mappa. Il primo bit di `doMove()` controllerà che il blocco verso il quale ci stiamo muovendo è all'interno dei limiti della nostra mappa. Questo è un modo spiccio di confrontare diversi valori e in termini semplici è 0 <= x < *width*, il che significa che x deve essere compreso tra 0 e la larghezza meno 1. Vedi figure4.py per il codice da usare.

Formattazione dei dati

Se stai usando dei dati codificati a mano, è una buona idea formattali in modo da poterli leggere facilmente

Definire la struttura dei dati del giocatore e disegnarne il personaggio sullo schermo

figure3.py

```
001. # Questo codice è vicino alla cima del nostro programma
002.
003. player = {"x":0, "y":3, "frame":0, "sx":0, "sy":0,
004.           "moveX":0, "moveY":0, "queueX":0, "queueY":0,
005.           "moveDone":True, "movingNow":False,
     "animCounter":0}
006.
007.
008. # Questo codice va nella funzione drawMap() all'interno
009. # del ciclo y
010.             if x == player["x"] and y == player["y"]:
011.                 if player["sx"] == 0:
012.                    player["sx"] = (x*32)-(y*32)+OFFSETX
013.                    player["sy"] = (y*16)+(x*16)+OFFSETY-32
014.                 screen.blit("ball"+str(player["frame"]),
     (player["sx"], player["sy"]))
015.
016.
```

figure4.py

```
001. def update(): # Funzione di update Pygame Zero
002.     global player
003.     if player["moveDone"] == True:
004.         if keyboard.left:
005.             doMove(player, -1, 0)
006.         if keyboard.right:
007.             doMove(player, 1, 0)
008.         if keyboard.up:
009.             doMove(player, 0, -1)
010.         if keyboard.down:
011.             doMove(player, 0, 1)
012.     updateBall(player)
013.
014. def doMove(p, x, y):
015.     if 0 <= (p["x"]+x) < mapInfo["width"] and 0 <=
     (p["y"]+y) < mapInfo["height"]:
016.         if mapData[p["x"]+x][p["y"]+y] == 0:
017             p.update({"queueX":x, "queueY":y,
     "moveDone":False})
018.
```

Catturare gli input dalla tastiera e impostare i dati per spostare il personaggio del giocatore

12 Occhio a dove metti i piedi

Dopo aver controllato che il movimento del giocatore sia all'interno dell'area della mappa, possiamo verificare se il movimento andrà su un blocco del pavimento (valore 0 nei nostri dati). Basterà solo controllare il valore in `mapData` e siamo a posto. la prossima riga di codice è un modo intelligente per cambiare diversi valori in un dizionario. Usiamo la funzione del dizionario `p.update()` per impostare i valori di `queueX`, `queueY` e `moveDone` tutti allo stesso tempo. Il motivo per cui accodiamo il movimento piuttosto che muovere direttamente è perché potremmo essere già in movimento e vogliamo aspettare il termine della mossa precedente.

> "Il motivo per cui accodiamo il movimento piuttosto che muovere è perché potremmo essere già in movimento"

Dizionari

Organizzare i dati in dizionari rende più facile capire per cosa sono usati i dati. È anche un buon passo verso l'utilizzo della programmazione orientata agli oggetti (OOP).

13 L'aggiornamento

In **figure4.py**, possiamo vedere come catturare dalla tastiera i controlli di movimento. Nella funzione `update()` di Pygame Zero, controlliamo se vengono premuti i tasti cursore e, in tal caso, richiamiamo la nostra funzione `doMove()` con gli adeguati parametri di movimento. Una volta controllato il movimento, richiamiamo una funzione `updateBall()` che si occuperà di tutto il pesante lavoro di animazione della palla e dello spostamento da un blocco all'altro. Noterai che prima di controllare la tastiera, ci accertiamo di essere pronti per ulteriori input controllando il valore del dizionario del giocatore `moveDone`, che abbiamo impostato come `False` in `doMove()`.

14 Sequenziare l'animazione

Tutto quello che ci resta da fare ora è far passare la palla da un blocco all'altro, ma se facciamo le cose nella sequenza sbagliata possiamo finire con la palla che viene disegnata di fronte ai blocchi quando dovrebbe essere dietro, o dietro i blocchi quando dovrebbe essere davanti. In questa fase possiamo cambiare i fotogrammi dell'animazione per fare rimbalzare la palla su e giù. Vogliamo anche che la palla si muova senza problemi da un posto all'altro, così più piccolo è il movimento da un `draw()` a quello successivo, e meglio è.

15 Rimbalziamo

Iniziamo con i fotogrammi dell'animazione della palla che rimbalza. Abbiamo otto frame chiamati da **ball0.png** fino a **ball7.png**. Se incrementiamo il valore del frame nel nostro dizionario giocatore ogni volta che chiamiamo `updateBall()` e quando arriviamo a 8 impostiamo il valore di nuovo a 0, la nostra funzione `drawMap()` si occuperà di disegnare l'animazione in un ciclo. Il solo il problema è che se eseguiamo l'animazione a questa velocità, la palla rimbalza troppo velocemente, e ci occorre rallentarla. Per questo usiamo un altro valore dal nostro dizionario del giocatore, `animCounter`. Con questo valore, contiamo ogni quattro fotogrammi e al quarto fotogramma aggiungiamo uno al valore `frame`.

16 Frame by frame

Abbiamo bisogno di tempo per far muovere la palla con i frame corretti in modo che sembri

rimbalzare senza problemi durante lo spostamento. Noi vogliamo attendere fino a `frame = 4` prima di iniziare lo spostamento. A quel punto spostiamo i valori di `queueX` e `queueY` in `moveX` e `moveY` e impostiamo `movingNow` come `True`. Quando `movingNow` è `True`, i valori `sx` e `sy` del giocatore sono cambiati. Per ciascuno blocco, abbiamo bisogno di spostare 32 pixel a sinistra o a destra con incremento di 1 pixel e 16 pixel in alto o in basso con Incrementi di 0,5 pixel. Quindi la nostra mossa impiegherà 32 cicli di aggiornamento.

17 Un blocco alla volta

Per mostrare la nostra palla in movimento correttamente nell'ordine di disegno della mappa, dobbiamo cambiare il blocco in cui si trova al frame corretto dell'animazione. quando `frame = 7` è il momento di cambiare, quindi a quel punto aggiorniamo i valori dei dizionari `x` e `y` del giocatore, basati sui valori `moveX` e `moveY`. Quindi impostiamo il valore `moveDone` del giocatore su `True`. Ciò significa che quando torneremo a `frame = 4`, possiamo cancellare `moveX` e `moveY`, e impostare `moveNow` su `False` a meno che un'altra mossa non sia stata accodata.

18 Avvolgendolo

Puoi vedere da figure5.py come funziona questa sequenza di frame nella funzione `updateBall()`. Vedrai prima di tutto il controllo per `movingNow` e usa una funzione separata, `moveP()`, per modificare le coordinate sullo schermo del giocatore. Quindi facciamo tutta la logica del sequenziamento azioni al fotogramma corrente. Vedrai anche un controllo se il labirinto è stato risolto. Abbiamo impostato una variabile globale, `mazeSolved`, su `False` all'inizio e se il giocatore arriva al blocco 11, 8 impostiamo la variabile su `True` e visualizziamo un messaggio adatto in `draw()`.

19 Livello uno completato

Si conclude la prima parte di questo tutorial. Abbiamo esaminato la creazione di una mappa dall'aspetto 3D da dati e immagini 2D e come spostare un personaggio animato sulla mappa. Questo formato di gioco ha molte possibilità e nel prossimo episodio vedremo come ingrandire la mappa, modificare i dati della mappa con un editor esterno e il caricamento dei dati da un file separato.

figure5.py

```
     def updateBall(p):
         global mazeSolved
         if p["movingNow"]:
004.         if p["moveX"] == -1: moveP(p,-1,-0.5)
005.         if p["moveX"] == 1: moveP(p,1,0.5)
006.         if p["moveY"] == -1: moveP(p,1,-0.5)
007.         if p["moveY"] == 1: moveP(p,-1,0.5)
008.     p["animCounter"] += 1
009.     if p["animCounter"] == 4:
010.         p["animCounter"] = 0
011.         p["frame"] += 1
012.         if p["frame"] > 7:
013.             p["frame"] = 0
014.         if p["frame"] == 4:
015.             if p["moveDone"] == False:
016.                 if p["queueX"] != 0 or p["queueY"] !=0:
017.                     p.update({"moveX":p["queueX"],
     "moveY":p["queueY"], "queueX":0, "queueY":0,
     "movingNow": True})
018.             else:
019.                 p.update({"moveX":0, "moveY":0,
     "movingNow":False})
020.                 if p["x"] == 11 and p["y"] == 8:
021.                     mazeSolved = True
022.
023.         if p["frame"] == 7 and p["moveDone"] == False and
     p["movingNow"] == True:
024.             p["x"] += p["moveX"]
025.             p["y"] += p["moveY"]
026.             p["moveDone"] = True
```